Vaccine Supervision

Progetto di Ingegneria del Software

Virginia Filippi e Chiara Solito

Corso di Laurea in Bioinformatica
Università degli studi di Verona
A.A. 2021/22

La presente è la documentazione del progetto del corso di Ingegneria del Software, svolto nel contesto della laurea triennale in bioinformatica.
Insieme a questo documento in formato PDF viene fornito anche il codice LaTeXcon cui è stato generato.

Contents

1 Traccia dell'Elaborato

Riportiamo di seguito la traccia dell'elaborato:

> Si vuole progettare un sistema software per gestire le segnalazioni di reazioni avverse (ad esempio, asma, dermatiti, insufficienza renale, miocardiopatia, ...) da vaccini anti-Covid.
> Ogni segnalazione è caratterizzata da un codice univoco, dall'indicazione del paziente a cui fa riferimento, dall'indicazione della reazione avversa, dalla data della reazione avversa, dalla data di segnalazione, e dalle vaccinazioni ricevute nei due mesi precedenti il momento della reazione avversa. Per ogni paziente sono memorizzati: un codice univoco, l'anno di nascita, la provincia di residenza e la professione.
> Per ogni paziente è possibile memorizzare gli eventuali fattori di rischio presenti (paziente fumatore, iperteso, sovrappeso, paziente fragile per precedenti patologie cardiovascolari/oncologiche), anche più d'uno. Ogni fattore di rischio è caratterizzato da un nome univoco, una descrizione e il livello di rischio associato. Per ogni paziente è, inoltre, memorizzata l'intera storia delle sue vaccinazioni precedenti, anti-Covid-19 e antinfluenzali.
> Ogni vaccinazione è caratterizzata da: paziente a cui si riferisce, segnalazioni a cui è legata, vaccino somministrato (AstraZeneca, Pfizer, Moderna, Sputnik, Sinovac, antinfluenzale, ...), tipo della somministrazione (I, II, III o IV dose, dose unica), sede presso la quale è avvenuta la vaccinazione e data di vaccinazione. Per ogni reazione avversa sono memorizzati un nome univoco, un livello di gravità (da 1 a 5) e una descrizione generale, espressa in linguaggio naturale. Una reazione avversa può essere legata a molte segnalazioni. Per ogni paziente sono memorizzati il numero di reazioni avverse segnalate ed il numero di vaccinazioni ricevute.
> Il sistema deve supportare i medici che effettuano la segnalazione. Dopo opportuna autenticazione, il medico viene introdotto ad una interfaccia che permette l'inserimento dei dati delle reazioni avverse e dei pazienti. Il codice univoco dei pazienti è gestito dal sistema, che tiene traccia dei pazienti indicati da ogni medico. Ogni medico vede solo i codici identificativi dei pazienti, dei quali ha già segnalato qualche reazione avversa, e le relative informazioni.
> Ad ogni fine settimana o quando il numero di segnalazioni raggiunge la soglia di 50, il sistema manda un avviso ad uno dei farmacologi responsabili della gestione delle segnalazioni di reazioni avverse. Il farmacologo, dopo autenticazione, accede alle segnalazioni (tutte, con l'indicazione del medico che le ha fatte) e può effettuare alcune analisi di base (quante segnalazioni per vaccino, quante segnalazioni gravi in settimana, quante segnalazioni per provincia e quante segnalazioni per sede di vaccinazione).
> Il sistema, inoltre, avvisa il farmacologo quando un vaccino ha accumulato in un mese oltre 5 segnalazioni di gravità superiore a 3. In base alle segnalazioni e agli avvisi del sistema, il farmacologo può proporre di attivare una fase di controllo del vaccino. Tale proposta viene registrata dal sistema, che tiene traccia di tutte le proposte relative ai vaccini segnalati.

2 Analisi e Specifica dei Requisiti

2.1 Specifiche casi d'uso

In questa sezione definiamo le proprietà dell'applicazione.
Come dichiarato nella traccia il sistema prevede l'utilizzo da parte di due tipologie di personale medico: Medico e Farmacologo. Entrambi i tipi di utente possono utilizzare l'applicazione dopo opportuno login: in questa sede si è previsto che gli utenti siano pre-registrati da un amministratore di sistema esterno (sul modello di sistemi medici già noti). Non è stato quindi previsto un form di registrazione, durante lo sviluppo e si suppone che ogni utente abbia a disposizione `username` e `password`.

Casi d'uso relativi al Medico

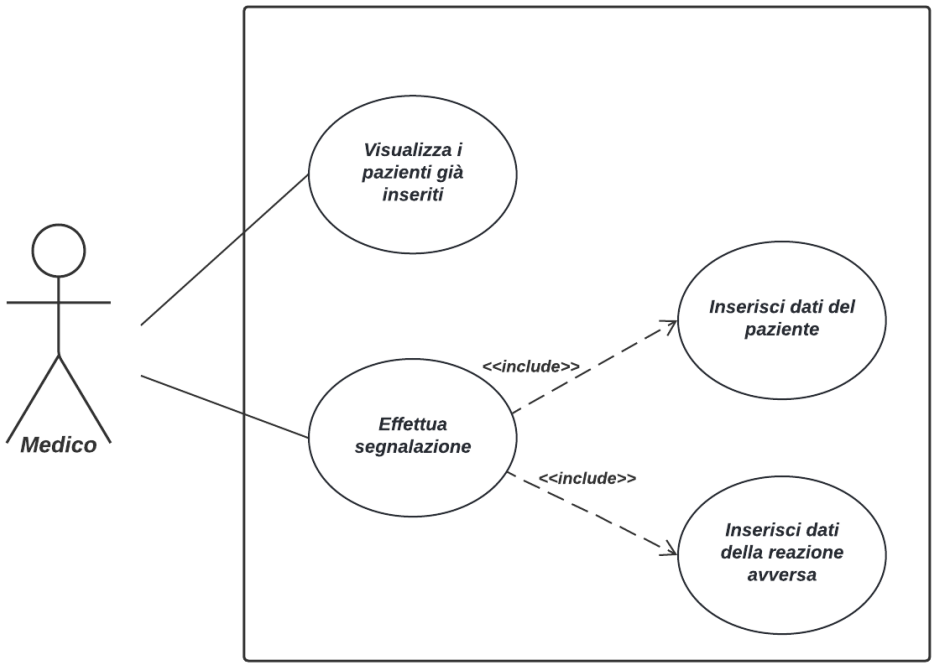

Figure 1: Caso d'uso Medico

Dopo opportuna autenticazione il medico viene introdotto all'interfaccia di base della sua sezione, che gli permette di:

- Visualizzare i pazienti già inseriti.
- Visualizzare i dati di un paziente, dalla lista dei pazienti già inseriti.
- Effettuare una segnalazione.

Visualizza i pazienti Una delle possibili attività che il Medico è autorizzato a fare è visualizzare la lista dei codici dei pazienti che lui stesso ha già inserito (gestiti automaticamente dal sistema), effettuando una segnalazione.

Caso d'uso: Visualizza i pazienti già inseriti
ID: UC1
Attori: Medico
Precondizioni: Il medico si è autenticato
Sequenza degli eventi: 1. Il medico accede al sistema 2. Il medico è introdotto all'interfaccia di base 3. Il medico accede all'interfaccia di visualizzazione dei pazienti segnalati 4. Il medico visualizza i pazienti da esso segnalati, con le relative informazioni
Postcondizioni: Nessuna

Figure 2: Caso d'uso UC1 del Medico

Il medico può visualizzare anche le informazioni relative ai pazienti di cui vede i codici:

Caso d'uso: Visualizza dati paziente
ID: UC2
Attori: Medico
Precondizioni: Il medico ha visualizzato la lista di pazienti segnalati
Sequenza degli eventi: 1. Il medico seleziona un paziente 2. Il medico seleziona l'opzione "Visualizza dati paziente" 3. Il sistema fornisce le informazioni base riguardo il paziente selezionato
Postcondizioni: Nessuna

Figure 3: Caso d'uso UC2 del Medico

Inseriamo ulteriori dettagli rispetto a questi due casi d'uso (gestiti in unico Sequence Diagram).

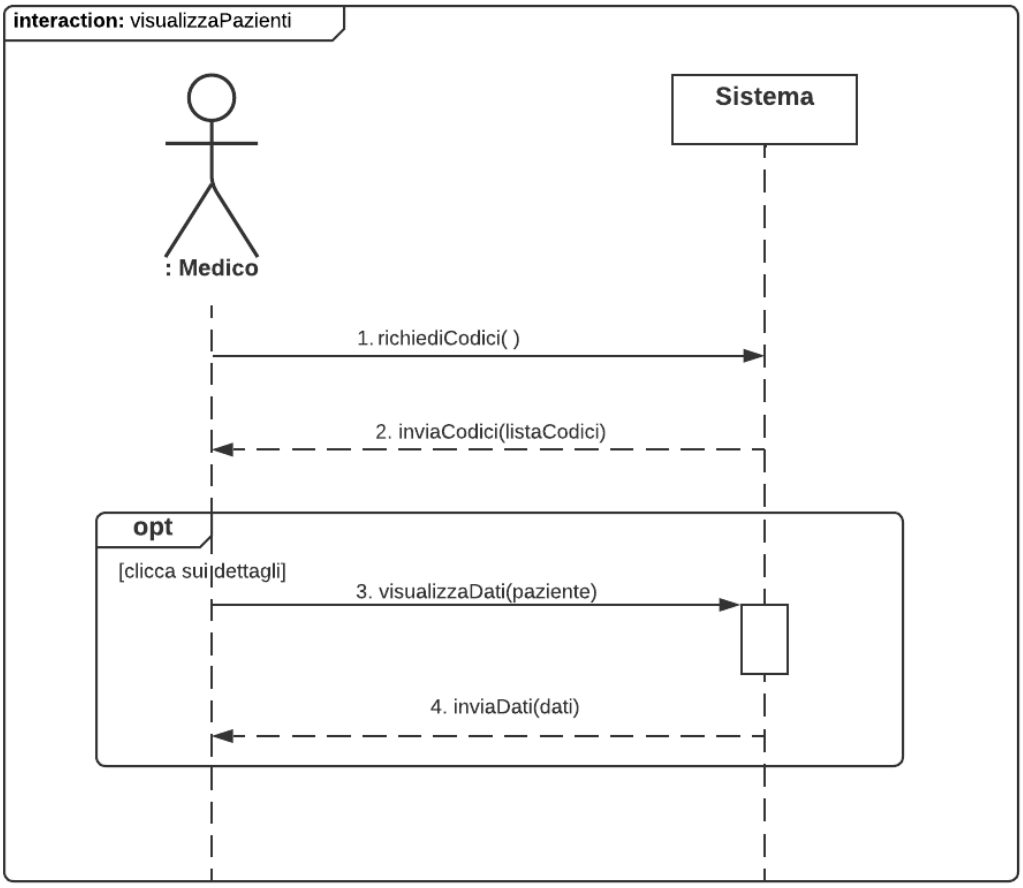

Figure 4: Sequence Diagram Visualizza Pazienti

Effettua segnalazione La principale attività dei medici è quella di inserire segnalazioni. Per fare questo, è necessario inserire i **dati del paziente** ed inserire i **dati della reazione avversa**. In questa sede si è scelto di dare uno specifico ordine a queste due azioni.
La data della segnalazione può essere gestita automaticamente oppure modificata dal medico.

Caso d'uso: Effettua segnalazione
ID: UC3
Attori: Medico
Precondizioni: Il medico si è autenticato
Sequenza degli eventi: 1. Il medico accede al sistema 2. Il medico è introdotto all'interfaccia di base 3. Il medico accede all'interfaccia per le segnalazioni 4. a) Il medico inserisce i dati relativi al nuovo paziente b) Il medico seleziona un paziente tra quelli a suo carico 5. a) Il medico inserisce una nuova reazione avversa b) Il medico seleziona una reazione avversa tra quelle già presenti nel sistema 6. Il medico inserisce la data della reazione avversa 7. Il medico conferma la segnalazione
Postcondizioni: La segnalazione è stata inserita

Figure 5: Caso d'uso UC3 del Medico

In questa fase prevediamo che il primo passo sia inserire i dati del paziente, abbiamo due alternative:

- Inserire nuovi dati del paziente, effettuando quindi l'intera procedura di inserimento (dati anagrafici, fattori di rischio e storia vaccinale)
- Scegliere uno dei pazienti già inseriti nel sistema.

L'identificativo univoco del paziente sarà gestito dal sistema.
Subito dopo si inseriscono i dati sulla reazione avversa. Anche in questa fase prevediamo le due alternative:

- Inserimento in sistema di una nuova reazione avversa, immettendo nome (univoco), gravità e descrizione.
- Selezione di una reazione avversa nota, pre-esistente nel sistema, aggiungendo la data.

Si prevede quindi che, data la storia vaccinale del paziente e la data della segnalazione, il sistema gestisce autonomamente le vaccinazioni del paziente entro i due mesi precedenti la segnalazione.

In ambo i casi, sarà necessario inserire la data della reazione avversa: questa dovrà essere consistente sia con la data della segnalazione sia con la data delle terapie del paziente.

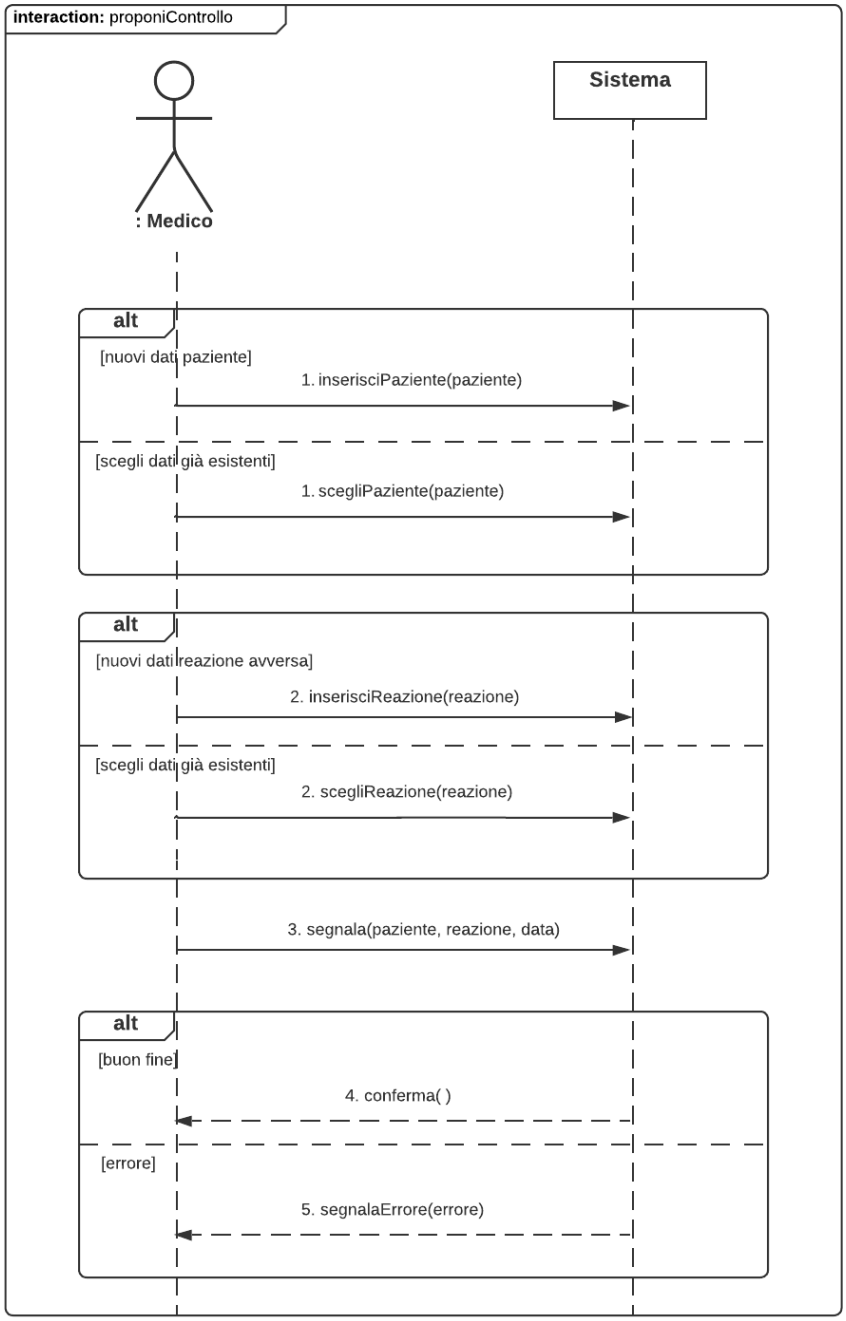

Figure 6: Sequence Diagram Effettua Segnalazione

Casi d'uso relativi al Farmacologo

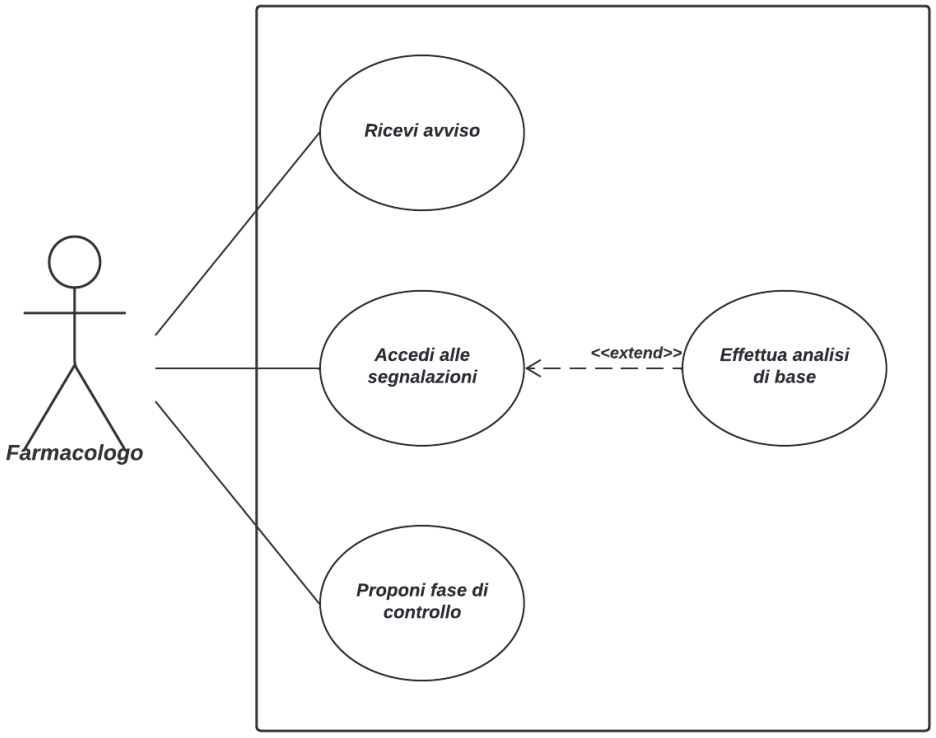

Figure 7: Caso d'uso Farmacologo

Dopo opportuna autenticazione il farmacologo viene introdotto all'interfaccia di base della sua sezione. Prima di poter effettuare qualsiasi azione gli vengono mostrati gli **avvisi non letti**. Dopo di che egli può:

- Accedere alle segnalazioni
- Dalle segnalazioni può effettuare delle analisi di base
- Proporre delle fasi di controllo

Avvisi Iniziamo dalla ricezione degli avvisi.
Il sistema deve fornire un meccanismo di gestione ed invio di avvisi verso i farmacologi, che si dividono in tre tipologie:

1. Il sistema manda un avviso non specifico se sono state raggiunte le 50 segnalazioni in una settimana.
2. Il sistema manda un avviso non specifico se è il weekend.
3. Il sistema manda un avviso specifico rispetto a un vaccino, se questo ha accumulato oltre 5 segnalazioni di gravità superiore a 3.

Il sistema avverte **tutti** i farmacologi responsabili. Ciò significa che ogni farmacologo all'autenticarsi riceverà gli avvisi generati e non ancora letti *da lui*: questo avviene in forma di pop-up e prima di vedere la schermata iniziale. È inoltre possibile, tramite un'opzione nel menù principale, rivedere gli avvisi già letti.

Caso d'uso: Ricevi avviso
ID: UC4
Attori: Farmacologo
Precondizioni: Il farmacologo si è autenticato, devono essere presenti avvisi nel sistema
Sequenza degli eventi: 1. Il farmacologo accede al sistema 2. Il farmacologo riceve gli avvisi non ancora letti
Postcondizioni: Gli avvisi devono essere segnati come già letti

Figure 8: Caso d'uso UC4 del Farmacologo

In fase di descrizione dei casi d'uso si è ritenuto opportuno inserire anche la possibilità di vedere gli avvisi già letti, sulla base dei quali il farmacologo propone le fasi di controllo:

Caso d'uso: Visualizza avvisi già letti
ID: UC5
Attori: Farmacologo
Precondizioni: Il farmacologo si è autenticato
Sequenza degli eventi: 1. Il farmacologo accede al sistema 2. Il farmacologo è introdotto all'interfaccia di base 3. Il farmacologo accede all'interfaccia di lettura degli avvisi già letti
Postcondizioni: Gli avvisi già letti sono visualizzati

Figure 9: Caso d'uso UC5 del Farmacologo

Leggi Segnalazioni Il farmacologo può accedere alla lista delle segnalazioni:

Caso d'uso: Accedi alle segnalazioni
ID: UC6
Attori: Farmacologo
Precondizioni: Il farmacologo si è autenticato
Sequenza degli eventi: 1. Il farmacologo accede al sistema 2. Il farmacologo è introdotto all'interfaccia di base 3. Il farmacologo accede all'interfaccia di controllo delle segnalazioni
Postcondizioni: Il farmacologo visualizza le segnalazioni

Figure 10: Caso d'uso UC6 del Farmacologo

Si è previsto che, come il medico visualizza informazioni di base dalla lista dei pazienti, anche il farmacologo possa visualizzare informazioni di base sulle segnalazioni.

Questo è stato specificato nel sequence diagram qui inserito:

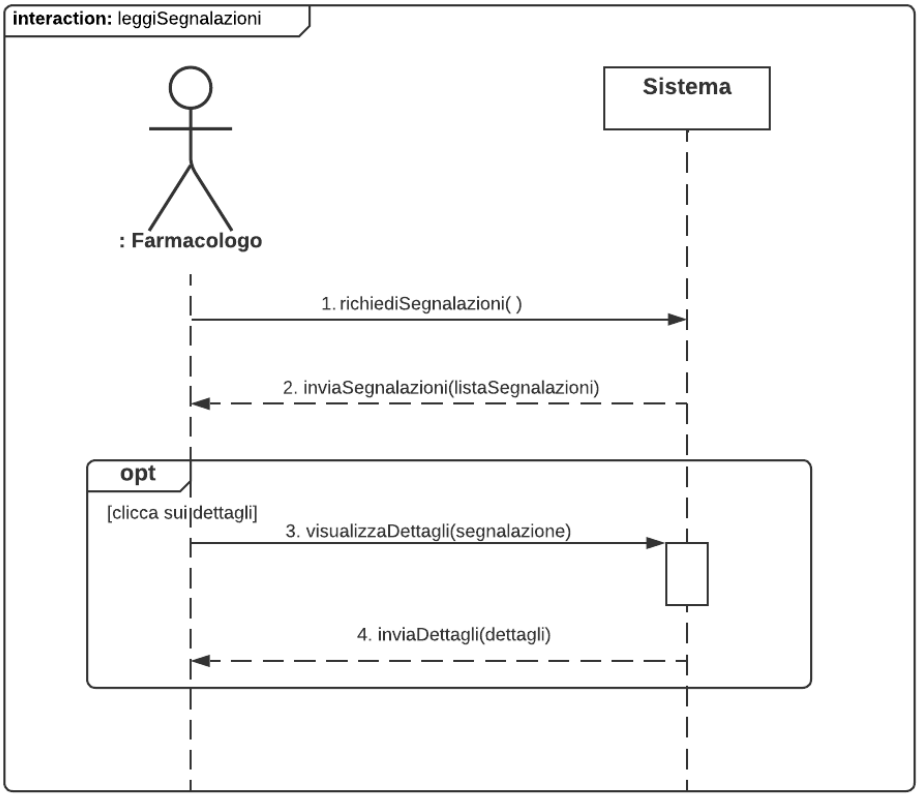

Figure 11: Sequence Diagram 1 Farmacologo

Effettua analisi di base Dalla lista delle segnalazioni può scegliere di effettuare delle **analisi di base**. Queste sono:

- Quante segnalazioni per vaccino
- Quante segnalazioni gravi in settimana
- Quante segnalazioni per provincia
- Quante segnalazioni per sede di vaccinazione

Caso d'uso: Effettua analisi di base
ID: UC7
Attori: Farmacologo
Precondizioni: Il farmacologo si è autenticato, deve aver visualizzato le segnalazioni
Sequenza degli eventi: 1. Il farmacologo seleziona l'opzione per le analisi 2. Il farmacologo seleziona l'analisi di interesse
Postcondizioni: Il sistema fa visualizzare i risultati dell'analisi richiesta

Figure 12: Caso d'uso UC7 del Farmacologo

Propone Fase di Controllo In base alle analisi eseguite e alla lettura delle segnalazioni, il farmacologo può proporre di attivare una fase di controllo del vaccino.
Tale proposta viene registrata dal sistema.

Caso d'uso: Propone fase di controllo
ID: UC8
Attori: Farmacologo
Precondizioni: Il farmacologo si è autenticato
Sequenza degli eventi: 1. Il farmacologo accede al sistema 2. Il farmacologo è introdotto all'interfaccia di base 3. Il farmacologo accede all'interfaccia per le proposte 4. Il farmacologo seleziona il vaccino 5. Il farmacologo dà la conferma
Postcondizioni: La proposta è memorizzata e tracciata dal sistema

Figure 13: Caso d'uso UC8 del Farmacologo

Di seguito un Sequence Diagram specifica queste azioni:

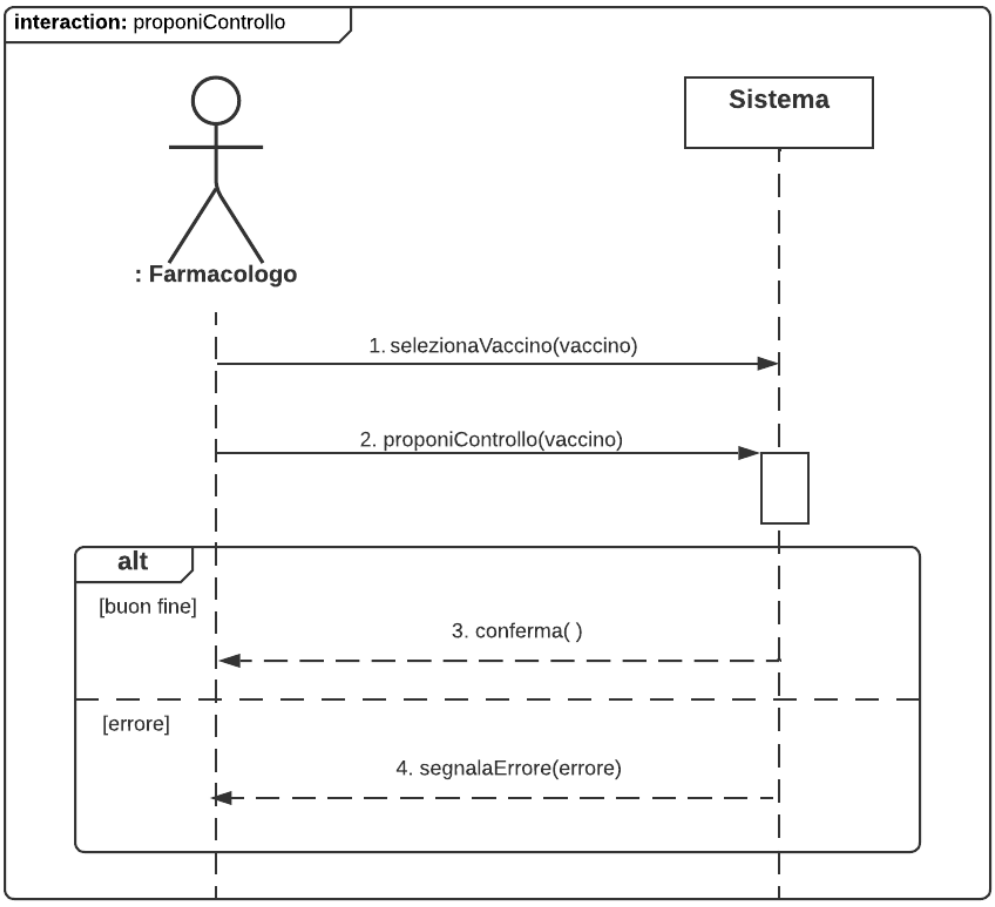

Figure 14: Sequence Diagram 2 Farmacologo

2.2 Diagrammi di Attività

Nota sui diagrammi Nei diagrammi relativi alle azioni successive al login non si è ritenuto di aggiungere una freccia che collega la scelta finale ad ogni interfaccia di partenza, per chiarezza e compattezza degli schemi.

Attività di Login

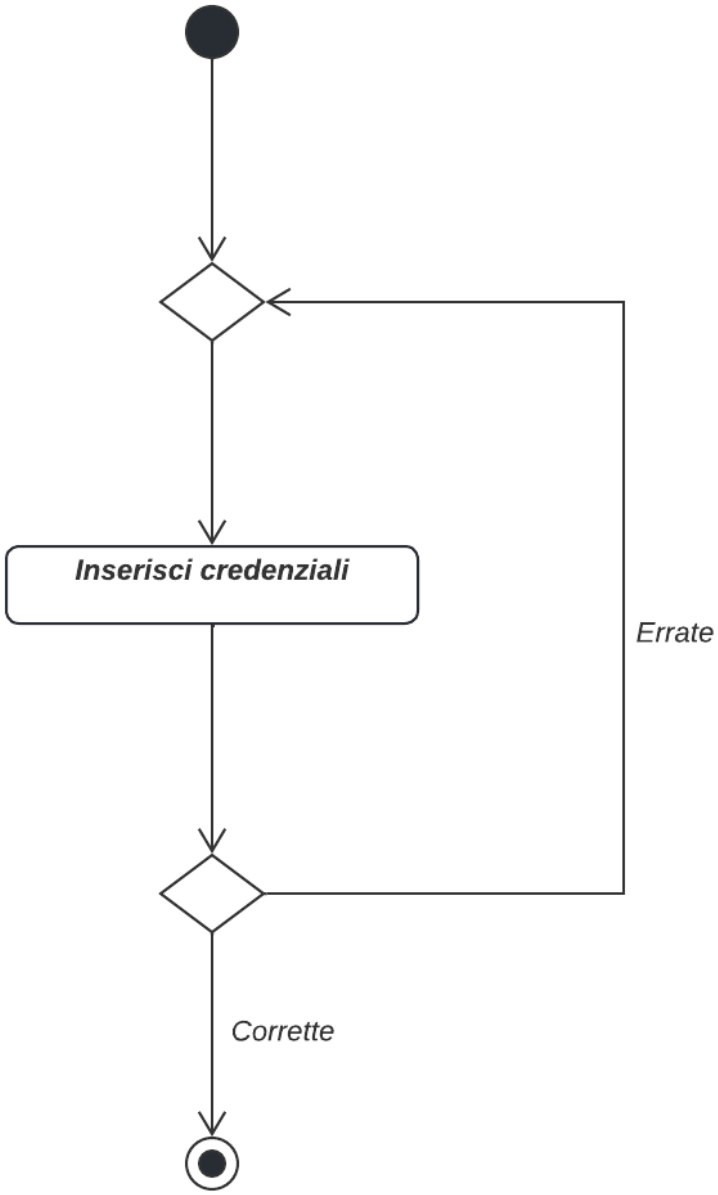

Figure 15

Attività dell'attore medico

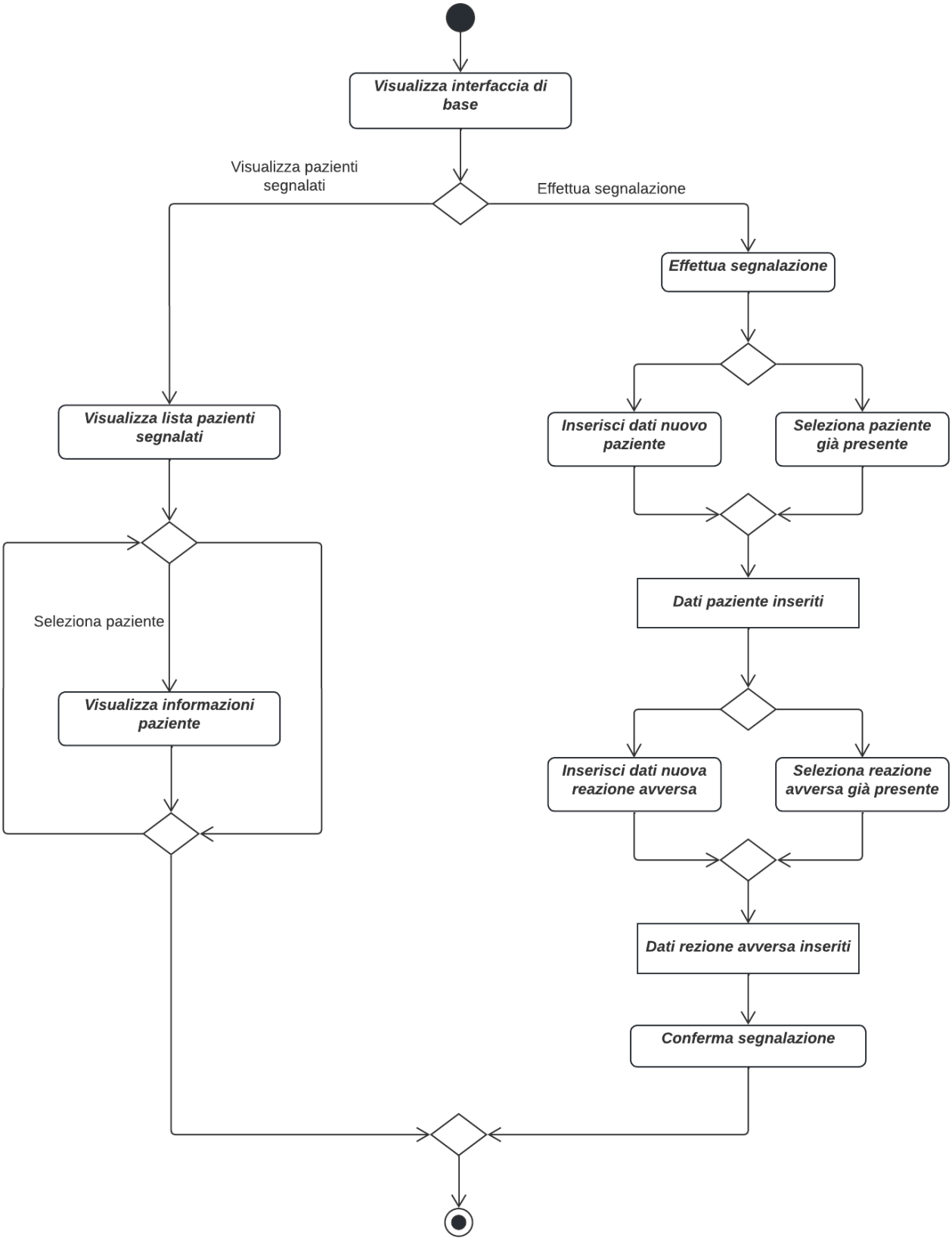

Figure 16

Attività dell'attore farmacologo

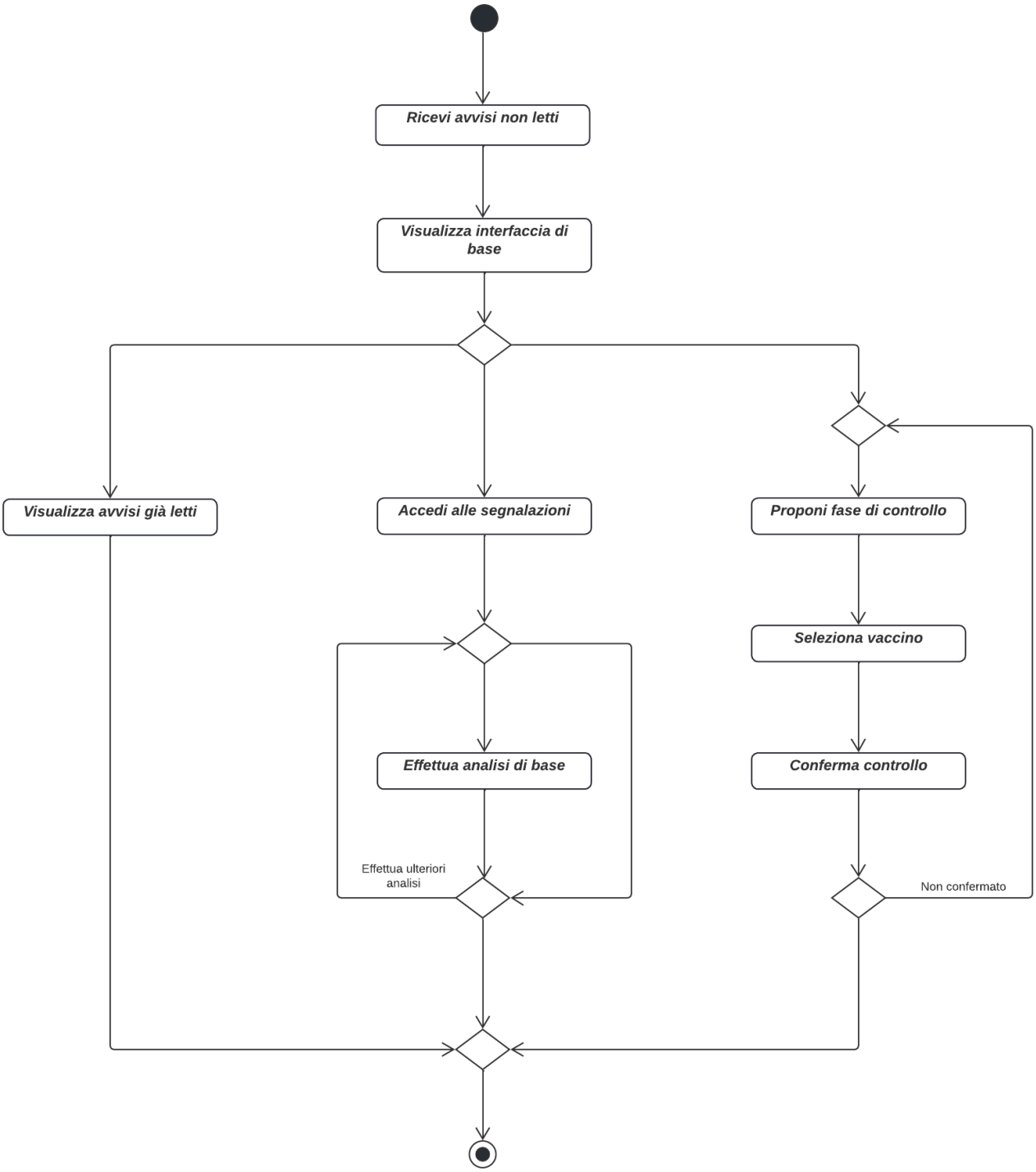

Figure 17

3 Scelte progettuali

3.1 Note sullo sviluppo

Si è scelto di sviluppare l'applicazione in Java, utilizzando la libreria **JavaFX**, senza l'ausilio di FXML, perché soddisfatti del design delle interfacce.
Il progetto è stato interamente sviluppato utilizzando **Git** come sistema di *code versioning* e GitHub per l'hosting della repository.
Abbiamo seguito una metodologia di sviluppo per lo più a cascata e **Plan Driven**: le fasi di specifica e di sviluppo sono state separate, cercando di porre comunque rimedio al principale svantaggio dello stesso, ovvero la difficoltà di cambiamento dopo aver avviato il processo. Questo è stato fatto utilizzando alcune metodologie proprie dello **sviluppo Agile**, soprattutto l'attività di *Pair Programming* che ha riguardato tutto lo sviluppo, con particolare attenzione allo sviluppo dei Modelli e delle prime View (si veda in seguito).
Ogni area è stata divisa in task in modo da alleggerire il carico di lavoro e i design pattern sono stati scelti nella fase di sviluppo in base a quelle che erano le nostre necessità, al fine di ottenere un codice performante ma allo stesso tempo leggibile.
Prima di cominciare lo sviluppo vero e proprio, si è condotta la fase di analisi dei requisiti che si è vista nella sezione precedente, generando i relativi use-cases e i diagrammi di attività. Anche la progettazione architetturale è stata fatta prima di iniziare il lavoro centrale, in modo da non avere problemi su quel frangente.

3.2 Pattern Architetturale - MVC

Abbiamo deciso di utilizzare il **pattern Model View Controller**, semplicemente perchè naturalmente implementabile con l'uso di JavaFX. Il sistema è strutturato in tre componenti logiche che interagiscono tra loro. Il **model** si occupa della gestione dei dati e delle operazioni su di essi. La componente **View** definisce le interfazze e come i dati vengono presentati all'utente. Il **Controller** gestisce infine l'interazione delle interfacce e dell'utente con i dati.

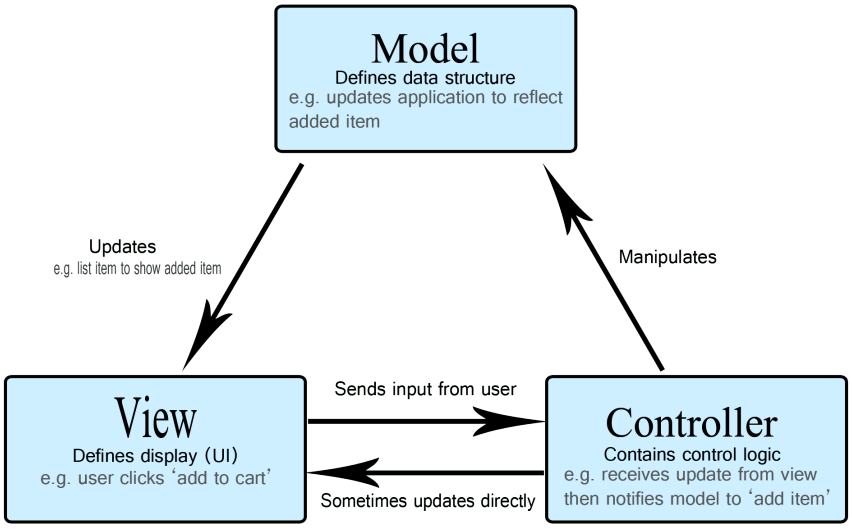

Figure 18: Pattern MVC

3.3 Data Access Object Pattern

Per la gestione dei dati si è ritenuto di utilizzare il pattern DAO, utilizzato per separare i dati di basso livello che accedono all'API dalle operazioni di alto livello.
Per ogni classe del Model (e di conseguenza per quasi tutte le tabelle implementate nel DataBase) si è implementato:

- **Interfaccia DAO** - Definisce le operazioni standard da eseguire.
- **Classe concreta DAO** - Implementa l'interfaccia di cuisopra. È responsabile dell'accesso ai dati dal database.
- **Model Object** - Questo è un semplice oggetto Java che contiene i metodi getter e setter per memorizzare i dati.

Le classi-modello implementate con il pattern DAO sono (in ordine alfabetico):

- **ControlPhase** - Fase di Controllo effettuata dal farmacologo.
- **Notice** - gli avvisi ricevuti dal farmacologo.
- **Patient** - i pazienti.
- **Reaction** - le reazioni avverse.
- **RiskFactor** - i fattori di rischio relativi ai pazienti.
- **Vaccination** - le vaccinazioni effettuate dai pazienti.

Nota Per la classe **User** (gli utenti che hanno accesso all'applicazione, ovvero dottori e farmacologi) si è decisso di non modellare la classe secondo il pattern DAO, ma di implementarla come un oggetto Java generico.

3.4 Diagrammi delle classi

Classi del Modello

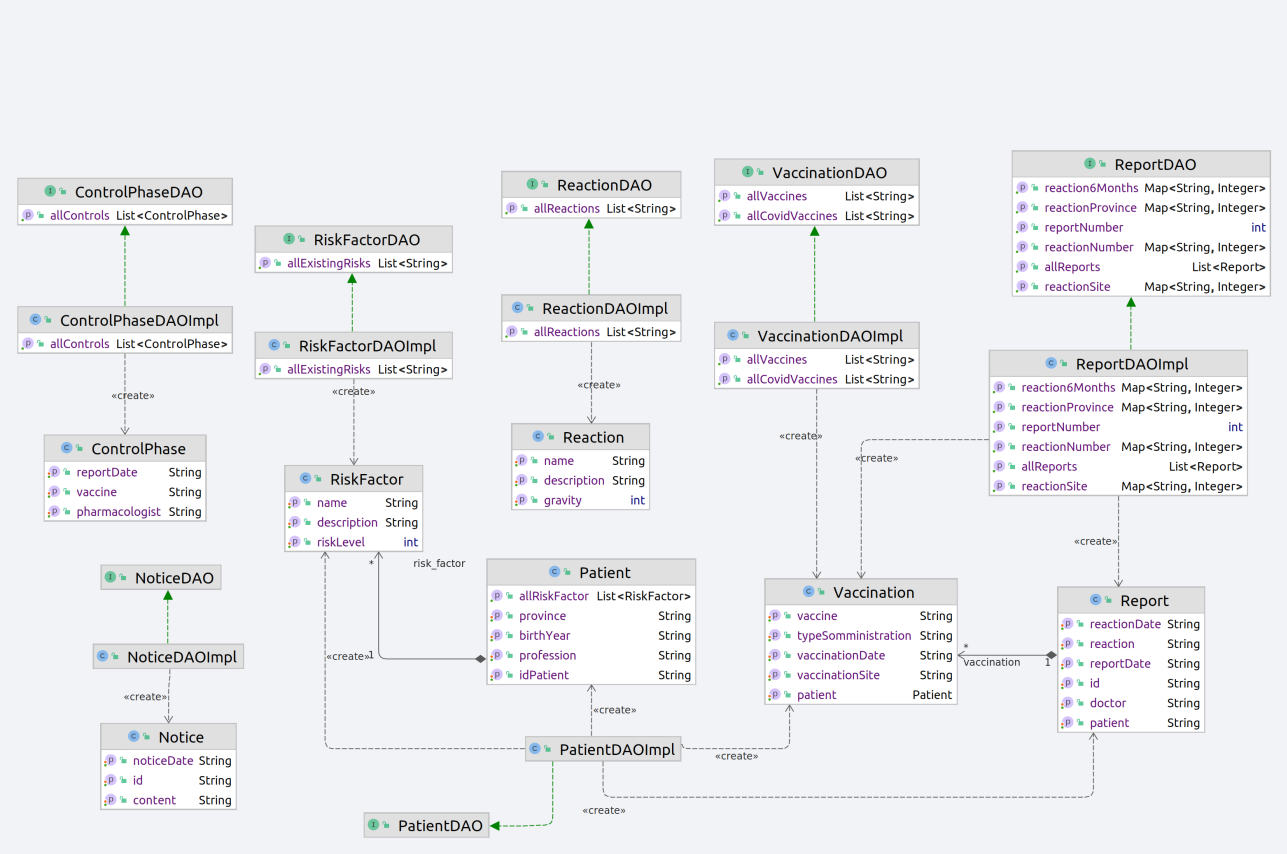

Figure 19: Interazione tra le classi del modello

Classi della View

Nota Non si sono inserite nel diagramma le classi considerate *Utils*, quindi le classi che ad esempio generano gli avvisi, i messaggi di errore o la lista dei vaccini possibili.

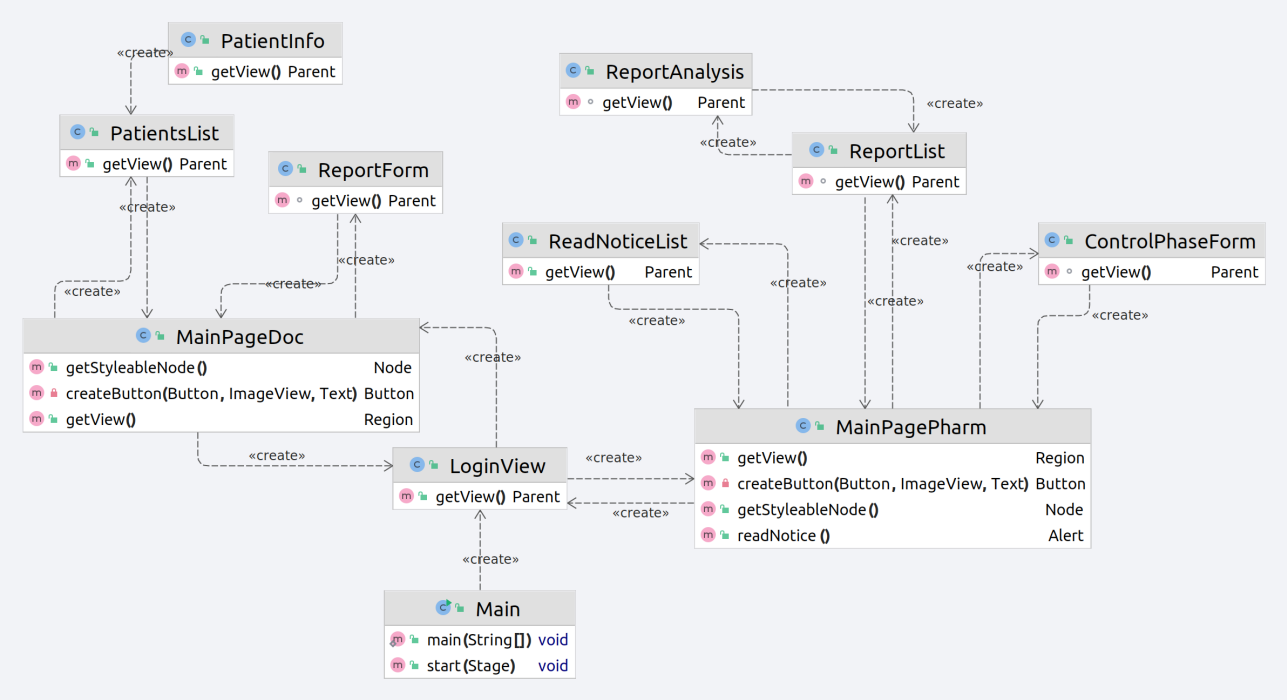

Figure 20: Interazione tra le classi della View

Esempio Interazione View - Controller

Nota Si è scelto di riportare come *Class Diagram* solo un esempio di **interazione tra Vista e Controllore**, perché esemplificativa delle interazioni generali tra questi due.
La vista chiama i metodi del controllore per poter interagire con i dati, a cui il controllore ha accesso (si veda la prossima DAO).

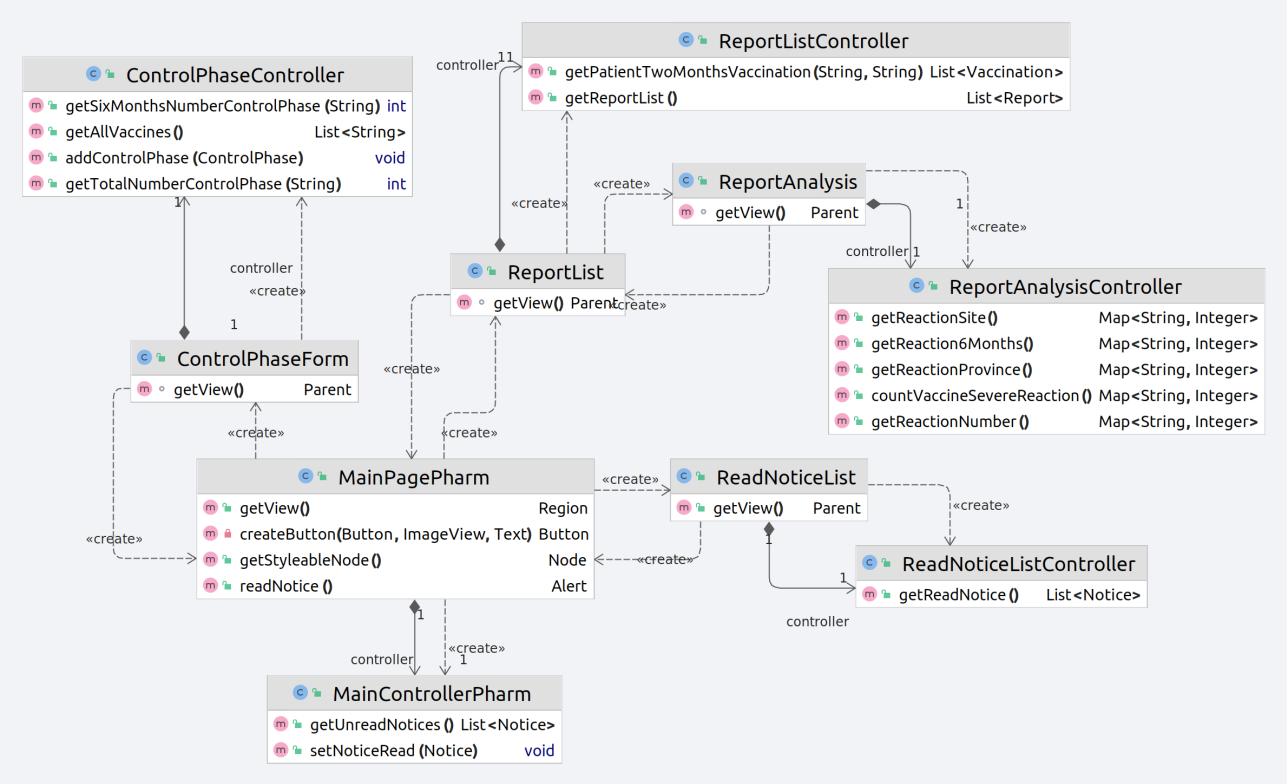

Figure 21: Interazione tra le classi della View Farmacologo e i loro controller

Esempio Interazione DAO - Controller

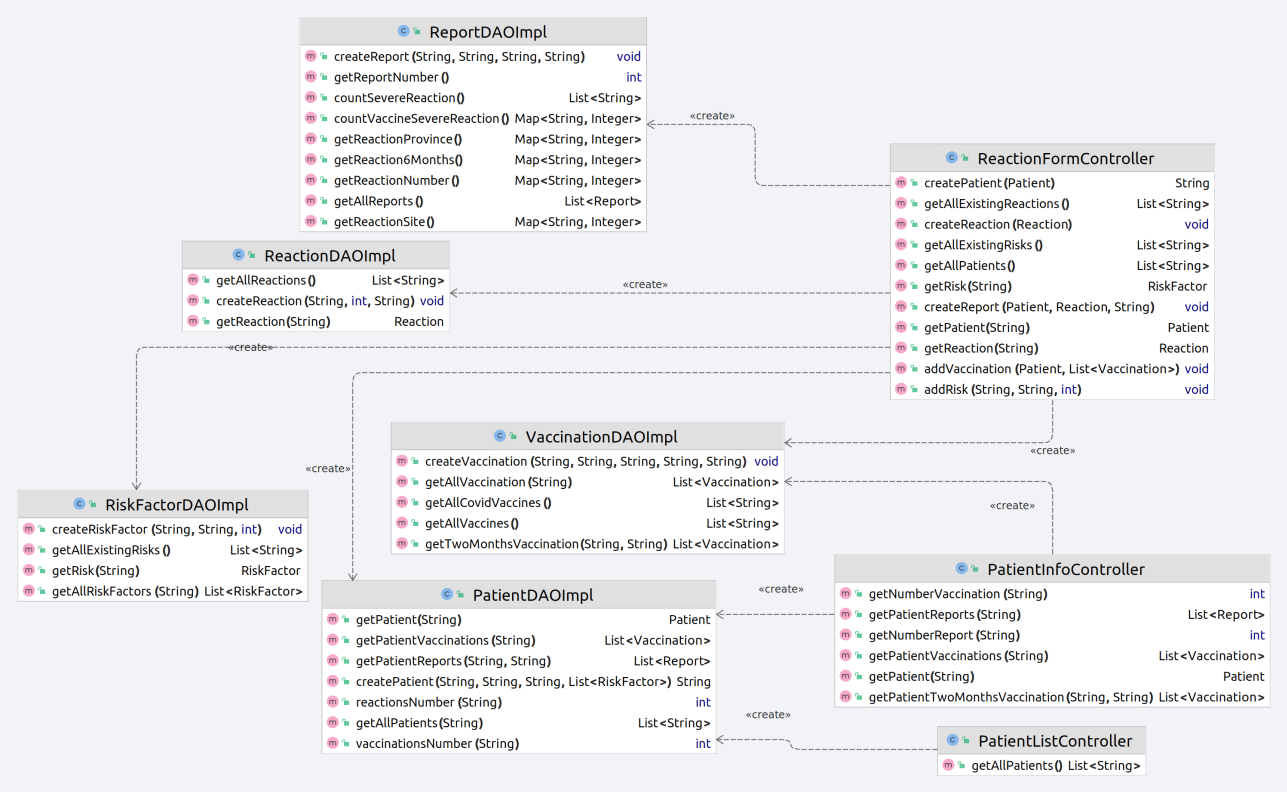

Figure 22: Interazione tra le classi del Controller del Dottore e i dati a cui ha accesso

4 Implementazione del DataBase

Come richiesto dalla traccia si è implementato un database con cui l'applicazione potesse interagire.
Il Database è stato creato sulla base dell'ER qui riportato:

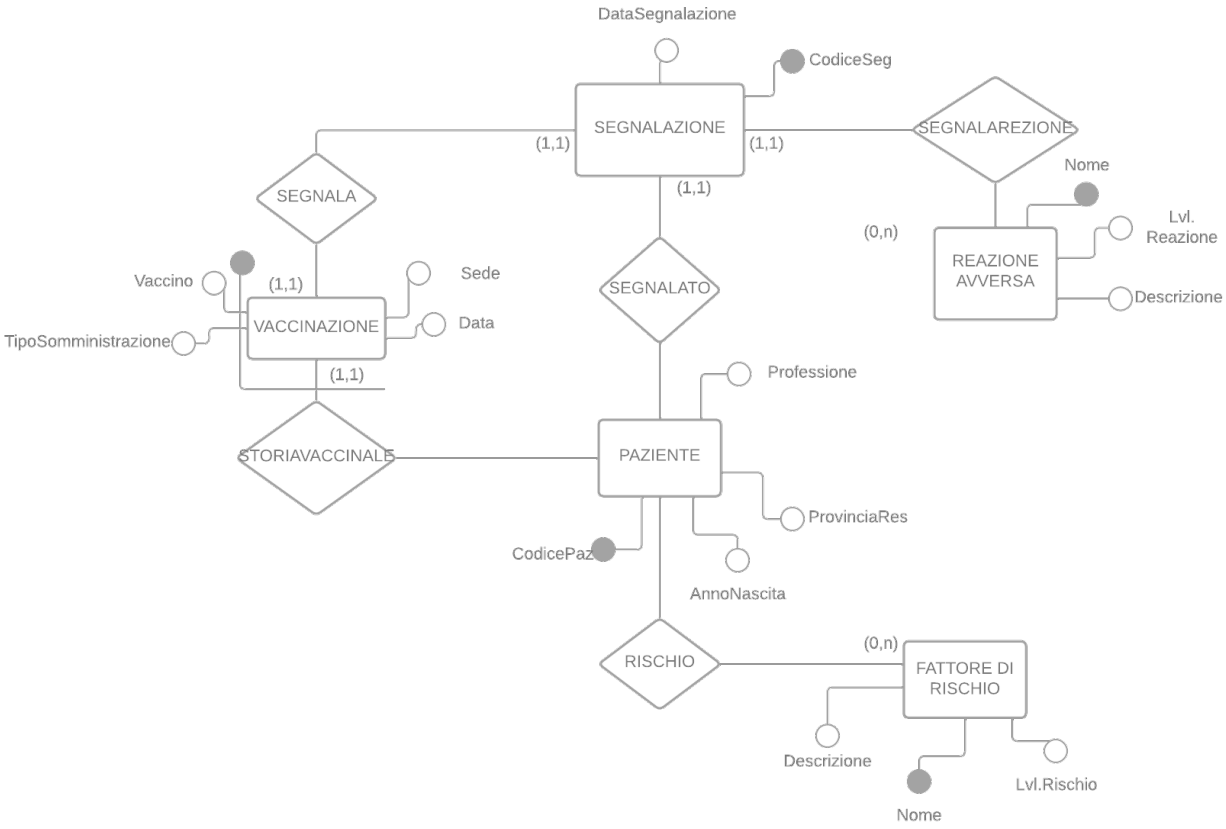

Figure 23: Schema ER da modificare!

5 Fase di Test

Il test dell'applicazione è stato iniziato in concomitanza con il termine del primo prototipo, in accordo con la metodologia di sviluppo Plan-Driven adottata.
Si è divisa la fase di test in 4 fasi principali:

1. Verifica interna della consistenza dell'applicazione rispetto all'analisi dei requisiti.
2. **Test di sviluppo**, implementati come unit test automatizzato mediante JUnit per verificare la correttezza di alcune delle classi più problematiche.
3. **Test di release**, consistente in alcuni stress test, con un team simulato di colleghi esterni allo sviluppo dell'applicazione.
4. **Test degli utenti**, in cui si è scelto un campione di conoscenti senza background informatico che verificassero il software facendo le veci di utenti reali in situazioni reali.

Nota

Il primo prototipo è da intendersi come la prima versione con tutte le features implementate ma senza aver passato il testing.

5.1 Test di Sviluppo: Unit Test

Per la parte di Test di sviluppo si è scelto di implementare degli Unit Test sulle classi del **Controller**. Questa scelta è derivata dal fatto che il controller rappresenta l'interazione tra l'interfaccia (testabile a livello visuale) e le DAO (le cui funzioni sono utilizzate dai controller e quindi implicitamente testate.)

- Per ogni calsse del controller ed ogni metodo si è scelto di **testare la normale funzione**: input normali con il confronto dei risultati rispetto a quelli che ci si aspetta dall'implementazione del DataBase utilizzata. Per questo la fase di testing unitario è stata statica e ha riguardato una sola giornata di lavoro, successivamente al termine del primo propototipo.
- Per ogni metodo si è implementato anche un test che verificasse l'*handling* degli errori all'interno delle classi. Dando in input oggetti o stringhe vuote si è verificato che i metodi non ritornassero alcuna informazione, o stampassero in **Standard Output** o **Standard Error** i messaggi di errore impostati nei catch delle eccezioni.

Tutte le classi di test hanno ottenuto un buon risultato, i metodi per cui sono stati rilevati errori (ad esempio non stampavano il messaggio esatto) sono state ricontrollate e corrette. Ogni test è visibile nella sezione apposita dell'eseguibile.

Appendice A: Query di creazione Database

Si è scelto di implementare il Database in PostgreSQL. Riportiamo anche le query usate per la creazione delle tabelle, che ci aiutano a comprendere com'è fatto:

Tabelle dello schema VaccineSupervisionDB

```
    CREATE TABLE patient(
        idPatient SERIAL PRIMARY KEY,
        birthYear NUMERIC(4) NOT NULL ,
        CHECK ( birthYear >= 1900 ),
        province VARCHAR(20) NOT NULL,
        profession VARCHAR(20) NOT NULL
);

CREATE TABLE RiskFactor(
    name VARCHAR(20) PRIMARY KEY,
    description VARCHAR(50),
    riskLevel NUMERIC(1) NOT NULL,
    CHECK ( riskLevel >= 1 AND riskLevel <= 5 )
);

CREATE TABLE Vaccination(
    idPatient INTEGER REFERENCES patient(idPatient),
    vaccine VARCHAR(15) NOT NULL,
    typeSomministration VARCHAR(10) NOT NULL,
    PRIMARY KEY (idPatient, vaccine, typeSomministration),
    vaccinationSite VARCHAR(10) NOT NULL,
    vaccinationDate DATE NOT NULL
);

CREATE TABLE Reaction(
    name VARCHAR(20) PRIMARY KEY,
    gravity NUMERIC(1) NOT NULL,
    CHECK(gravity >= 1 AND gravity <= 5),
    description VARCHAR(50) NOT NULL
);

CREATE TABLE PatientRisk(
    idPatient INTEGER NOT NULL REFERENCES patient(idPatient),
    risk VARCHAR(20) NOT NULL REFERENCES RiskFactor(name),
    PRIMARY KEY(idPatient, risk)
);

CREATE TABLE users(
    username VARCHAR(10) PRIMARY KEY,
    password VARCHAR(12) NOT NULL,
    type BOOLEAN NOT NULL,
    UNIQUE (username, password)
);

CREATE TABLE Report(
    id SERIAL PRIMARY KEY,
    reportDate DATE NOT NULL DEFAULT CURRENT_DATE,
    reactionDate DATE NOT NULL,
    reaction VARCHAR(20) NOT NULL REFERENCES Reaction(name),
```

```
    idPatient INTEGER NOT NULL REFERENCES patient(idPatient),
    doctor VARCHAR(10) REFERENCES users(username)
);

CREATE TABLE ControlPhase(
    pharmacologist VARCHAR(10) REFERENCES users(username),
    vaccine VARCHAR(15) NOT NULL,
    PRIMARY KEY (pharmacologist, vaccine)
);
```